Hidden Markov Model Baseline for Flexible Assembly System Anomaly Detection - DRAFT

Tero Keski-Valkama

Version: January 3, 2022

***Abstract*—**

***Index Terms*—**

I. Introduction

Flexible Assembly Systems (FAS) and Flexible Manufacturing Systems (FMS) represent an evolution from specialized mass manufacturing systems. They allow flexibility in manufacturing which enables efficient production of smaller production runs and customization, personalization and quick evolution of manufactured products.

Flexibility brings challenges as it makes fault detection harder, especially in systemic conditions. Even if each separate module of a flexible assembly plant was nominally operating well, there are systemic fault modes and degradations which can cause non-optimal performance of the whole. Novel production plans deployed to the manufacturing system evokes new behaviours and possible problems. As flexible manufacturing generally requires more open integration to external systems for planning and controlling production dynamically, it becomes more susceptible to cyberattacks.

Many of these fault conditions and degradations are such that they cannot be trivially known in advance. Industrial anomaly detection systems are used to detect novel anomalous conditions which might denote faults, degradations or for example cyberattacks. Traditional process mining methods are not well suitable for environments with heterogeneous industrial IoT systems creating logs where process trace events might not be fully correlated with the process ids. These uncorrelated process traces or logs are therefore interlaced so that events are in sequence but originating from separate processes. In general case, automatic process identification might not be even possible in an explicit form.

Machine learning approaches can provide automatic learning of industrial process nominal operation, and those can be used to flag anomalous situations. Machine learning systems require data to train on, and industrial process data is guarded trade secret especially in relation to fault conditions. Real data on manufacturing process faults is nontrivial to get in sufficient amounts and on system level. Hence, simulations for synthetic flexible assembly process are required. In this research, FAS Simulator is used as the data source [1].

To compare different machine learning approaches, baselines are needed. We are interested in anomaly detection solutions for uncorrelated process traces, or sequential, interlaced event logs. Existing methods for these include Hidden Markov Model based approaches [2] [3], denoising autoencoder based approaches [4], LSTM based approaches [5] [6], or even more complex deep neural networks [7]. Some of the traditional approaches such as Alpha algorithm from process mining require correlated process traces, where each event is tagged with the process instance it belongs to, effectively deinterlacing the event logs.

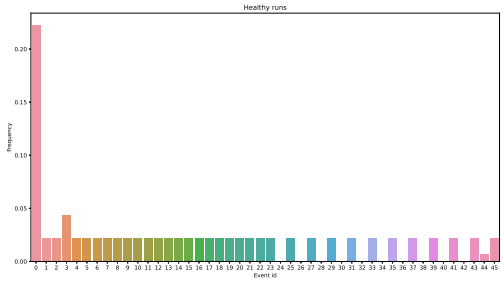

Fig. 1. Frequencies of events in healthy runs. The event id zero corresponds to the TICK event.

Here we present results of applying Hidden Markov Models on FAS Simulator datasets to produce a baseline benchmark for this challenge to compare more sophisticated approaches against.

II. Setup

We generated challenge data using FAS Simulator project [8]. The challenge dataset consists of 10,000 runs of healthy flexible assembly system and 10,000 runs of a flexible assembly system under a randomly picked degrading condition. Each run consists of assembling 30 items, slightly over 1,000 events logged each.

The event ids are enumerated in the Table I.

The data is divided into training set of the first 90,000 healthy runs, and the validation set of the final 10,000 healthy runs and all the 100,000 degraded runs. For training, 1,000 runs are picked from the training set randomly, and for validation 1,000 runs are picked from the validation set randomly.

Two types of degradations are simulated for validation: Manual step retry delay for a random bowl feeder, and a wear and tear degradation for a random conveyor belt. For eventual testing, a new type of degradation will be simulated.

64 Hidden Markov Models are trained with different numbers of hidden states from 1 to 64. Different lengths of sequential windows are used to train and evaluate the model, with window sequence lengths of 10, 100, 1,000 and full runs.

The trained HMM models are used to score the overall likelihood of the observed sequence window assuming the trained model for the validation set sample sequences.

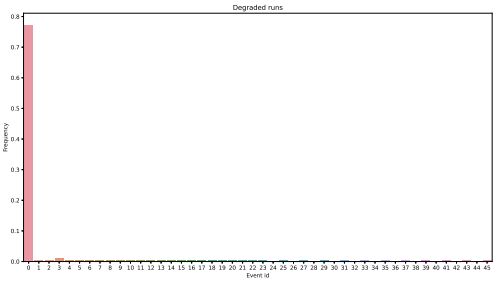

Fig. 2. Frequencies of events in degraded runs. The event id zero corresponds to the TICK event.

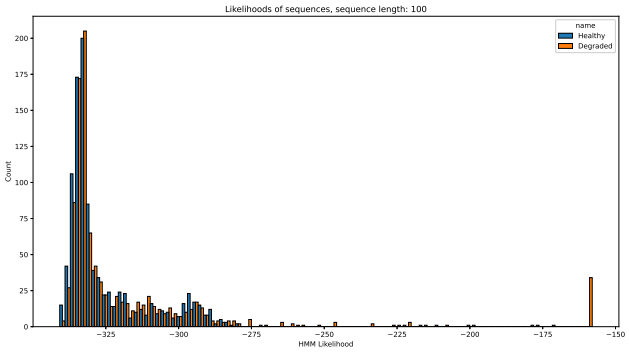

Fig. 3. Counts of different HMM log likelihood scores for both healthy and degraded runs, sequence length 100.

Plotting simple HMM log likelihood score histograms for different log likelihood values (higher log likelihood implies a higher likelihood the sequence was generated by an equivalent process to the trained HMM) shows that there is significant overlap for healthy and degraded runs, and that certain degraded runs seem to get higher log likelihood scores than healthy runs. See figure 3. This implies that certain degraded runs are a better match to the trained HMM than healthy runs in the validation set.

Fig. 4. Counts of different HMM log likelihood scores for both healthy and degraded runs, full sequences.

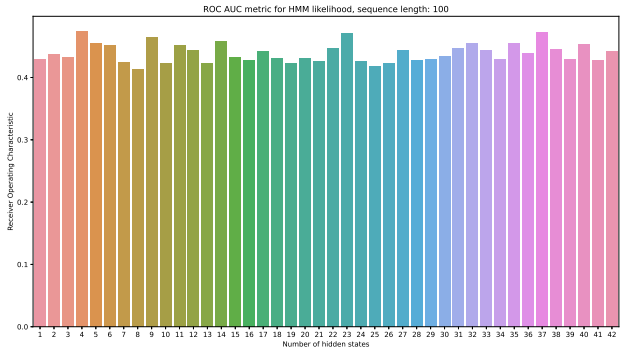

Fig. 5. Receiver Operating Characteristic AUC scores for different numbers of hidden states used using HMM log likelihood scores with a sequence length of 100.

TABLE I
TABLE OF THE EVENT IDENTIFIERS

Event id	Event name
0	TICK
1	CONVEYOR1 CONVEYOR_GATE
2	CONVEYOR2 CONVEYOR_GATE
3	CONVEYOR3 CONVEYOR_GATE
4	CONVEYOR4 CONVEYOR_GATE
5	CONVEYOR5 CONVEYOR_GATE
6	CONVEYOR6 CONVEYOR_GATE
7	CONVEYOR7 CONVEYOR_GATE
8	CONVEYOR8 CONVEYOR_GATE
9	CONVEYOR9 CONVEYOR_GATE
10	CONVEYOR10 CONVEYOR_GATE
11	CONVEYOR11 CONVEYOR_GATE
12	CONVEYOR_INPUT_SUBASSEMBLY_B CONVEYOR_GATE
13	CONVEYOR_INPUT_SUBASSEMBLY_C CONVEYOR_GATE
14	BOWL1 GIVEN
15	BOWL2 GIVEN
16	BOWL3 GIVEN
17	BOWL4 GIVEN
18	CRANE1 FORWARD
19	CRANE1 BACKWARD
20	CRANE1 STOP
21	CRANE_INPUT_SUBASSEMBLY_A FORWARD
22	CRANE_INPUT_SUBASSEMBLY_A BACK-WARD
23	CRANE_INPUT_SUBASSEMBLY_A STOP
24	MANUAL_INSPECTION QUEUE_ALARM
25	MANUAL_INSPECTION OK
26	MANUAL_ADD_COMPONENTS1 QUEUE_ALARM
27	MANUAL_ADD_COMPONENTS1 OK
28	MANUAL_ADD_COMPONENTS2 QUEUE_ALARM
29	MANUAL_ADD_COMPONENTS2 OK
30	MANUAL_COMBINE_SUBASSEMBLY_A QUEUE_ALARM
31	MANUAL_COMBINE_SUBASSEMBLY_A OK
32	MANUAL_COMBINE_SUBASSEMBLY_B QUEUE_ALARM
33	MANUAL_COMBINE_SUBASSEMBLY_B OK
34	MANUAL_ADD_COVER_AND_BOLTS QUEUE_ALARM
35	MANUAL_ADD_COVER_AND_BOLTS OK
36	MANUAL_TIGHTEN_BOLTS1 QUEUE_ALARM
37	MANUAL_TIGHTEN_BOLTS1 OK
38	MANUAL_COMBINE_SUBASSEMBLY_C QUEUE_ALARM
39	MANUAL_COMBINE_SUBASSEMBLY_C OK
40	MANUAL_TIGHTEN_BOLTS2 QUEUE_ALARM
41	MANUAL_TIGHTEN_BOLTS2 OK
42	MANUAL_ADD_COMPONENTS3 QUEUE_ALARM
43	MANUAL_ADD_COMPONENTS3 OK
44	MANUAL_TIGHTEN_BOLTS3 QUEUE_ALARM
45	MANUAL_TIGHTEN_BOLTS3 OK

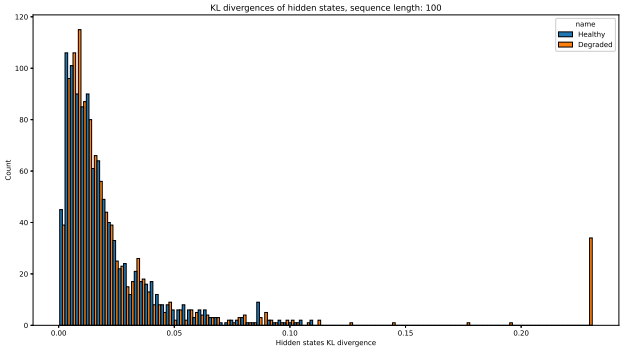

Fig. 6. Counts of different HMM hidden state KL divergences for both healthy and degraded runs, sequence length 100.

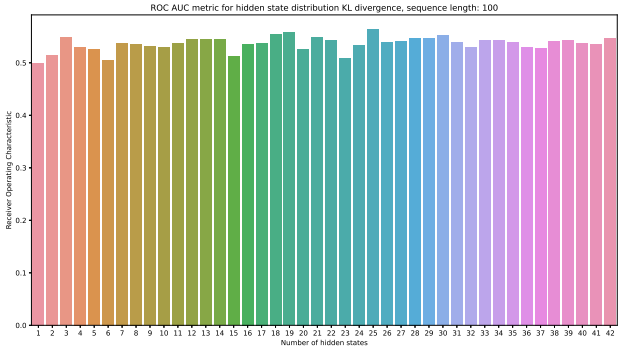

Fig. 7. Receiver Operating Characteristic AUC scores for different numbers of hidden states used using HMM hidden state KL divergence scores with a sequence length of 100.

This is likely due to the fact that HMM is unable to perfectly model the causal relationships in the process traces, and that degraded sequences tend to have a higher frequency of "TICK" events (Figure 2) in the traces designating the passing of time than in healthy sequences (Figure 1). As "TICK" events are the most common event type in the training set as well, their respective probability is very high compared to other event types. This makes the HMM score healthy logs lower than degraded logs.

When we graph the ROC AUC scores based on these log likelihoods across all the different numbers of hidden states (Figure 5) we notice that the values are less than 0.5 when the full sequence is not used. This means that this model has negative predictiveness and we would do better if we inverted the predictions of the model to swap prediction to healthy when the model predicts degraded, and vice versa. However, it is not likely that this swap will generalize well to new fault types, so we will need to predict anomalous divergences differently.

We also notice that the number of hidden states doesn't seem to have a great effect.

Using full sequences the ROC AUCs show values over 0.5, possibly because using the full sequences has the added characteristic that every sequence starts from the same state of the flexible assembly system, namely empty state, while windowed sequences are from random offsets in a complete assembly run. See figure 3.

Other research using HMMs for anomaly detection has found that naive scoring of the sequences doesn't work well for this purpose [2]. Instead, they tend to suggest looking at the HMM hidden state sequences and differences of those to the healthy hidden state sequences.

In our case, the event ids are static across sequences, and the trained HMM model is static as well, so we can simply compare the hidden state sequences directly without the added complication of equivalence or similarity of different Hidden Markov Models. We simply compute histograms of hidden states in the observed window for the trained HMM, and compare the histograms to the histograms of the training set using KL divergence.

Doing the same for KL divergences (0 means perfect distributional match of the observed HMM hidden state distribution to the training data hidden state distribution, and larger values denote worse match) shows that some of the degraded runs do indeed have worse KL divergences than the KL divergences for the healthy runs. See Figure 6.

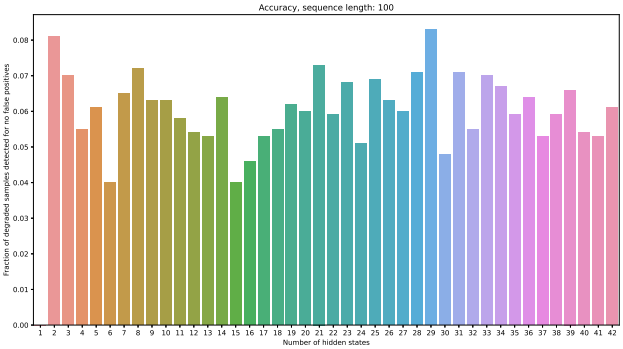

Fig. 8. KL divergence score based correct discrimination fraction for degraded samples for no false positives with a sequence length of 100.

When we graph the ROC AUC scores based on these KL divergences across all the different numbers of hidden states (Figure 7), we notice that the number of hidden states doesn't seem to have a great effect here either as long as there are at least 3 hidden states.

The discrimination rate for the model when the cut-off score is set to the maximal divergence for the healthy samples, that is, for no false positives, is 6% and at most 8% (Figure 8).

REFERENCES

[1] T. Keski-Valkama, "A simulator for event-oriented data in flexible assembly system fault prediction," *Procedia computer science*, vol. 119, pp. 121–130, 2017.

[2] N. Görnitz, M. Braun, and M. Kloft, "Hidden markov anomaly detection," in *International conference on machine learning*. PMLR, 2015, pp. 1833–1842.

[3] S. S. Joshi and V. V. Phoha, "Investigating hidden markov models capabilities in anomaly detection," in *Proceedings of the 43rd annual Southeast regional conference-Volume 1*, 2005, pp. 98–103.

[4] T. Nolle, A. Seeliger, and M. Mühlhäuser, "Unsupervised anomaly detection in noisy business process event logs using denoising autoencoders," in *International conference on discovery science*. Springer, 2016, pp. 442–456.
[5] L.-P. Yuan, P. Liu, and S. Zhu, "Recompose event sequences vs. predict next events: A novel anomaly detection approach for discrete event logs," in *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, 2021, pp. 336–348.
[6] M. Du, F. Li, G. Zheng, and V. Srikumar, "Deeplog: Anomaly detection and diagnosis from system logs through deep learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1285–1298.
[7] X. Zhang, Y. Xu, Q. Lin, B. Qiao, H. Zhang, Y. Dang, C. Xie, X. Yang, Q. Cheng, Z. Li *et al.*, "Robust log-based anomaly detection on unstable log data," in *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2019, pp. 807–817.
[8] T. Keski-Valkama, "Flexible assembly simulation: FAS Simulator." [Online]. Available: https://github.com/keskival/FAS-Simulator

Tero Keski-Valkama Tero Keski-Valkama is working as a lead ML & AI engineer in HERE Switzerland GmbH. He has been programming neural networks since high school in 1990s.